

(54) PROGRAM LANGUAGE TRANSLATOR

(11) 3-85639 (A) (43) 10.4.1991 (19) JP

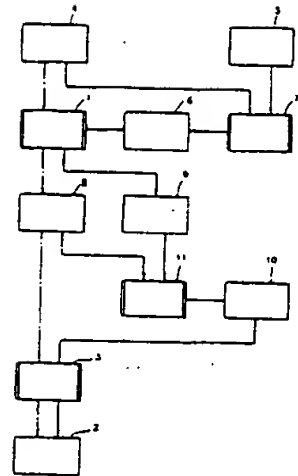
(21) Appl. No. 64-221573 (22) 30.8.1989

(71) FUJITSU LTD (72) HIROTOSHI YAMADA(4)

(51) Int. Cl. G06F9/45

**PURPOSE:** To shorten time required for source analysis, and to improve the efficiency of translation processing by executing the source analysis only for a corrected part at the time of the translation of a source program after correction.

**CONSTITUTION:** After the correction of the source program, differential information 6 between the source program 4 before correction and the source program 5 after correction is delivered to a source analyzing means 1 by a hysteresis information control means 7. Then, a differential intermediate code 9 corresponding to the differential information 6 is substituted for a part corresponding to the corrected part of an intermediate code 8 before correction generated and preserved last time by the source analyzing means 1 by a substituting means 11. The substituted intermediate code 10 is delivered to a code generating means 3, and an object program 2 equivalent in a meaning to the source program 5 after correction is generated. Thus, when the source program is corrected and altered, since the source analysis is executed only for the corrected part, the source analysis need not be executed for the whole source program, and the processing time is shortened, and the processing efficiency is improved.



**This Page Blank (uspto)**

⑨ 日本国特許庁(JP)

⑩ 特許出願公開

## ⑫ 公開特許公報(A)

平3-85639

⑬ Int.Cl.<sup>5</sup>

識別記号

庁内整理番号

⑭ 公開 平成3年(1991)4月10日

G 06 F 9/45

8724-5B

G 06 F 9/44

3 2 2 E

審査請求 未請求 請求項の数 1 (全 10 頁)

⑮ 発明の名称 プログラム言語翻訳機

⑯ 特 願 平1-221573

⑰ 出 願 平1(1989)8月30日

⑱ 発 明 者 山 田 博 敏 神奈川県川崎市中原区上小田中1015番地 富士通株式会社内

⑲ 発 明 者 池 田 功 神奈川県川崎市中原区上小田中1015番地 富士通株式会社内

⑳ 発 明 者 中 沢 通 太 神奈川県川崎市中原区上小田中1015番地 富士通株式会社内

㉑ 発 明 者 山 本 浩 福岡県福岡市博多区博多駅前1丁目5番1号 富士通九州通信システム株式会社内

㉒ 出 願 人 富士通株式会社 神奈川県川崎市中原区上小田中1015番地

㉓ 代 理 人 弁理士 松 本 昂

最終頁に続く

## 明 細 書

## 1. 発明の名称

プログラム言語翻訳機

## 2. 特許請求の範囲

入力原始プログラムを解析し、命令単位の間  
コードに展開するソース解析手段(1)と、

各中間コードに対応する命令列をオブジェクト  
プログラムとして生成するコード生成手段(3)と、

修正前原始プログラム(4)と修正後原始プログラ  
ム(5)との差分情報(6)を前記ソース解析手段(1)に渡  
す履歴情報管理手段(7)と、

前記ソース解析手段(1)により生成された修正前  
原始プログラム(4)に応じた修正前中間コード(8)の  
修正箇所に対応する部分を、前記ソース解析手段  
(1)により生成された差分情報(6)に応じた差分中間  
コード(9)に置換えて、前記コード生成手段(3)に渡  
す置換手段(10)とを具備してなることを特徴とする  
プログラム言語翻訳機。

## 3. 発明の詳細な説明

## 概 要

プログラム言語で記載された原始プログラムを  
実行形式のプログラムに翻訳するプログラム言語  
翻訳機に関し、

翻訳処理時間が短く、処理効率の高いプログラ  
ム言語翻訳機の提供を目的とし、

入力原始プログラムを解析し、命令単位の間  
コードに展開するソース解析手段と、各中間コー  
ドに対応する命令列をオブジェクトプログラムと  
して生成するコード生成手段と、修正前原始プロ  
グラムと修正後原始プログラムとの差分情報を前  
記ソース解析手段に渡す履歴情報管理手段と、前  
記ソース解析手段により生成された修正前原始プ  
ログラムに応じた修正前中間コードの修正箇所  
に対応する部分を、前記ソース解析手段により生  
成された差分情報に応じた差分中間コードに置換  
えて、前記コード生成手段に渡す置換手段とを具  
備して構成する。

産業上の利用分野

本発明はプログラム言語で記載された原始（ソース）プログラムを実行形式のプログラムに変換するプログラム言語翻訳機に関する。

プログラム言語（ソース言語）で記載された原始プログラムを、実在する計算機の上で実行させるためには、その目的計算機が実行できる形式のプログラム（オブジェクトプログラム）に、予め変換しておかなければならない。このために用いられるのがプログラム言語翻訳機であり、このプログラム言語翻訳機は、一般に原始プログラムを文法に従って解析する解析モジュールと、この解析データから原始プログラムと意味の等価なオブジェクトコードを生成するコード生成モジュール等からなる。これらのモジュールはさらにサブモジュールに分解することができ、前段のモジュールの処理結果を一旦、中間コードとしてファイルに格納し、これを次段の入力として処理する多重バス形式のものがある。

このようなプログラム言語翻訳機においては、

- 3 -

要素に分けられ、構文解析部ではこの基本構成要素からプログラムの構造を決定し、文法に合致するかが確かめられ、コード生成に必要な情報が取り出される。これらの処理を経た中間コード45はファイルに格納される。

次いで、この中間コード45は最適化処理部43に入力される。最適化処理部43では、実行時間の短縮化や記憶領域の縮小化等を目的とした最適化処理がなされ、この結果は中間コード46としてファイルに格納されるとともに、コード生成部44に入力される。

コード生成部44では、入力された情報に基づいてプログラム中の変数や作業場所に対して実行時の記憶場所が決定され、構文解析の結果判明したプログラムの意味と、記憶割当の方針に従って、オブジェクトプログラム47を生成し、問題が無ければオブジェクトプログラム47をファイルに格納して処理を終了するようになっている。

プログラムにバグが発見される等してプログラムの修正が必要になった場合には、従来はプログ

例えば数万行の原始プログラムに対して数行の修正を加えた場合に、修正を施したプログラム全体を再翻訳する必要があるため、時間的ロスが大きく、プログラム作成上の大きな障害となっている。このため、このような障害を除去し、効率的な翻訳処理を実現できるプログラム言語翻訳機の提供が要望されている。

従来の技術

第5図は従来のプログラム言語翻訳機の一例を示すブロック図である。40はプログラマにより作成され、ファイルに格納されている原始プログラムであり、この原始プログラム40は目的計算機で実行可能な形式のプログラムに変換するため翻訳機41に入力される。翻訳機41はソース解析部42、最適化処理部43、及びコード生成部44から構成されている。ソース解析部42はさらに字句解析部と構文解析部に分けることができ、字句解析部では入力された原始プログラム40が名前、定数、特殊記号等のプログラムの基本構成

- 4 -

ラムエディタにより原始プログラムを修正し、翻訳機41により前回の処理と同様の処理を再度実施してオブジェクトプログラムを生成していた。

発明が解決しようとする課題

しかし、従来技術によると、例えば、原始プログラムを数時間費やして翻訳処理してオブジェクトプログラムを生成した後、軽微なバグにより数行の修正を加えたような場合であっても、再度同じ処理を数時間かけて実施する必要があり、これは非常に効率が悪く、プログラム作成上の大きな障害となっていた。

本発明はこのような点に鑑みてなされたものであり、翻訳処理時間が短く、処理効率の高いプログラム言語翻訳機の提供を目的としている。

課題を解決するための手段

第1図は本発明の原理を説明するための図であり、本発明を構成する各手段とこれらの手段間でのデータの流れが示されている。同図中、一点鎖

- 5 -

- 6 -

縦矢印は新規に作成された原始プログラムに対する翻訳処理を、実線矢印は修正が加えられた原始プログラムに対する翻訳処理を示している。

本発明のプログラム言語翻訳機は、入力された原始プログラムを解析し、命令単位の間中コードに展開するソース解析手段1と、中中コードに対応する命令列をオブジェクトプログラム2として生成するコード生成手段3と、修正前原始プログラム4と修正後原始プログラム5との差分情報6をソース解析手段1に渡す履歴情報管理手段7と、ソース解析手段1により生成された修正前原始プログラム4に応じた修正前中中コード8の修正箇所に対応する部分を、ソース解析手段1により生成された差分情報6に応じた差分中中コード9と置換えて修正後中中コード10とし、これをコード生成手段3に渡す置換手段11とから構成されている。

#### 作 用

本発明によるプログラム言語翻訳機は、新規に

- 7 -

に渡して、コード生成手段3により修正後原始プログラム5と意味の等価なオブジェクトプログラム2が生成される。

このように、本プログラム言語翻訳機によれば、原始プログラムが修正変更された場合には、その修正された部分のみについてソース解析を行い、前回翻訳時に生成された修正前中中コードの修正箇所に対応する部分をこれと置換えるようにしているから、2回目以降の翻訳時には原始プログラム全体についてソース解析を実施する必要が無く、その処理時間を大幅に短縮することができる。

#### 実 施 例

以下本発明の実施例を図面に基づいて詳述する。

第2図は本発明の一実施例を説明するためのブロック図であり、(A)は新規に作成した原始プログラムに対しての翻訳処理を、(B)は1回以上の修正を加えた原始プログラムに対しての翻訳処理を示している。

20はプログラムエディタを有する世代管理シ

- 9 -

作成された原始プログラムに対して翻訳処理(1回目の翻訳処理)を行うときには、第1図中、一点鎖線矢印で示されているように、ソース解析手段1により原始プログラム4全体についての中中コード8を生成してこれを保存しておくとともに、コード生成手段3に渡してオブジェクトプログラム2を得る。

バグ対策等により原始プログラムを修正した場合(2回目以降の翻訳処理)においては、第1図中、実線矢印で示されているように、履歴情報管理手段7により、修正前原始プログラム4と修正後原始プログラム5の差分情報6がソース解析手段1に渡され、1回目(前回)ソース解析手段1により生成され保存された修正前中中コード8の修正箇所に対応する部分を、この差分情報6に対応する差分中中コード9と置換手段11により置換える。この置換えられた中中コード10は、この時点で、修正後原始プログラム5全体をソース解析手段1で解析して生成された中中コードと同一のものとなっている。これをコード生成手段3

- 8 -

ステムであり、プログラマはこの世代管理システム20のプログラムエディタを用いて原始プログラム21、22を作成することができる。また、この世代管理システム20は、バグ対策等のためにプログラムを修正した場合に、履歴情報管理ファイル23に修正変更した情報等が自動的に記録されるようになっており、プログラムの履歴を管理する機能を有している。

本プログラム言語翻訳機は、この世代管理システム20により作成された原始プログラム21、22を翻訳処理するものであり、新規に作成した原始プログラム21と、これに1回以上の修正を加えた原始プログラム22とで、その処理内容が異なっている。

まず、新規に作成された原始プログラムに対する処理を第2図(A)を参照して説明する。世代管理システム20のプログラムエディタにより、新規に作成され、ファイルに格納されている原始プログラム21は目的計算機で実行可能なプログラムに変換するために翻訳機24に入力される。

- 10 -

原始プログラム 21 は、例えば第 3 図 (A) に示されるように記述されたプログラムである。尚、第 3 図 (A) 中の行番号はこの言語の文法上記述されているものではない。

新規な原始プログラム 21 に対しての本翻訳機の機能的な構成は第 2 図 (A) 中、点線内に示されているとおりである。即ち、翻訳機 24 はソース解析部 25、最適化処理部 26、及びコード生成部 27 から構成されている。ソース解析部 24 はさらに字句解析部と構文解析部に分けることができ、字句解析部では入力された原始プログラム 21 が名前、定数、特殊記号等のプログラムの基本構成要素に分けられ、構文解析部ではこの基本構成要素からプログラムの構造を決定し、文法に合致するか否かが確かめられ、コード生成に必要な情報が取り出される。これらの処理を経た中間コード (ソース解析終了時の中間コード) 28 はファイルに格納される。中間コード 28 は例えば、第 3 図 (B) に示されるような内容のものであり、中間コードはテキスト、行番号、辞書、及び修正

情報から構成されている。修正情報は後述する最適化処理において辞書に変更を生じた場合にその情報を記録する部分であり、この段階では使用されていない。

次いで、この中間コード 28 は最適化処理部 26 に入力される。この最適化処理部 26 では、実行時間の短縮化や記憶領域の縮小化等を目的とした最適化処理がなされ、この結果は最適化処理終了時の中間コード 29 としてファイルに格納され保存される。この中間コード 29 は例えば、第 3 図 (C) に示されるような内容のものである。同図 (B) と比較して明らかなように、100 行目の辞書 (A=B L@) を削除し、替わって (A/=B L@) を追加するとともに、300 行目の L1 を削除することにより最適化が図られている。尚、ソース解析終了時の中間コード 28 は最適化処理終了時の中間コード 29 のファイル化と前後してファイル上から削除される。

次いで、この最適化処理終了時の中間コード 29 は、コード生成部 27 に渡される。コード生成

- 1 1 -

- 1 2 -

部 27 では、中間コード 29 の情報に基づいてプログラム中の変数や作業場所に対して実行時の記憶場所が決定され、構文解析の結果判明したプログラムの意味と、記憶割当の方針に従って、オブジェクトプログラム 30 が生成され、このオブジェクトプログラム 30 がファイルに格納されて翻訳処理が終了する。

次に、第 2 図 (B) を参照して、2 回目以降の翻訳処理について説明する。プログラムにバグが発見される等してプログラムの修正が必要になった場合には、世代管理システム 20 のプログラムエディタにより前回作成した新規な原始プログラム 21 の所定箇所を修正して原始プログラム 22 を作成する。この原始プログラム 22 は例えば、第 4 図 (A) に示されるように記述されたものであり、100 行目と 500 行目に修正が加えられている。この修正を実施したときに、世代管理システム 20 はこの修正内容を履歴情報として履歴情報管理ファイル 23 に格納している。

そして、この修正後の原始プログラム 22 の翻

訳処理を実施するときには、翻訳機 24 には履歴情報管理ファイル 23 に格納されている修正された内容 (以下、差分情報) と、この原始プログラムが新規のものでないという情報が入力される。翻訳機 24 ではこの原始プログラムが新規のものでないという情報を受けて以下のような処理を実施する。尚、各部の処理について第 2 図 (A) を参照して既に説明した部分についての説明は一部省略する。

即ち、差分情報はソース解析部 24 で解析されて差分中間コード 31 としてファイルに格納される。差分中間コード 31 は例えば第 4 図 (B) に示されるような内容である。次いで、この差分中間コード 31 及び前回の翻訳処理において作成された中間コード (最適化処理がなされたもの) 29 が中間言語再生成部 32 に入力される。

中間言語再生成部 32 では、前回の翻訳処理で作成された最適化終了時の中間コード 29 (第 3 図 (C)) から、これに含まれている修正情報 (最適化処理による修正情報) を基に最適化処理を実

- 1 3 -

- 1 4 -

施する前の中間コード28(第3図(B))を復元する。そして、この復元された中間コード28の修正箇所に対応する部分(100行目と500行目)を、差分中間コード31(第4図(B))に置換えて、これを修正後中間コード33としてファイルに格納する。この中間コード33は例えば、第4図(C)に示されるような内容のものである。第3図(B)と比較して100行目と500行目が差分中間コード31に置換えられていることがわかる。この時点で、修正後中間コード33は修正された原始プログラム22全体についてソース解析を実施した場合と同一の結果となっている。

次いで、修正後中間コード33は最適化処理部26に渡され、前回翻訳処理時と同様な最適化処理がなされて、例えば、第4図(D)に示されるような中間コードに変換された後、コード生成部27に入力され、オブジェクトプログラム30が生成されて翻訳処理が終了する。

2回以上の修正があった場合には、この2回目

の翻訳処理と同様の処理が繰り返し実行される。

上述したように、本実施例によれば、原始プログラムに修正が加えられた場合に、このプログラムに対する翻訳処理においては、原始プログラム全体に対してソース解析をすることがなく、修正された箇所のみについてソース解析を実施した後、前回翻訳時の中間コードの修正箇所に対応する部分をこれと置換えるようにしている。これにより、ソース解析に要する処理時間は修正箇所の大きさに依存することになり、プログラム全体をソース解析しないから、修正後原始プログラムの翻訳に要する処理時間が大幅に短縮される。

#### 発明の効果

本発明は以上詳述したように、修正後原始プログラムの翻訳時には、修正箇所以外の部分についてソース解析を実施しないように構成したから、ソース解析に要する処理時間が大幅に短縮され、翻訳処理の効率を大幅に向上することができるという効果を奏する。

- 15 -

- 16 -

#### 4. 図面の簡単な説明

第1図は本発明の原理構成を示すブロック図、

第2図(A)及び(B)は本発明の一実施例を示すブロック図であり、(A)は新規に作成した原始プログラムに対しての翻訳処理を示す図、

(B)は修正した原始プログラムに対しての翻訳処理を示す図である。

第3図(A)～(C)は新規に作成した原始プログラム及び1回目の翻訳処理時に出力される中間コードの一例を示す図、

第4図(A)～(D)は修正された原始プログラム及び2回目以降の翻訳処理時に出力される中間コードの一例を示す図、

第5図は従来技術を示すブロック図である。

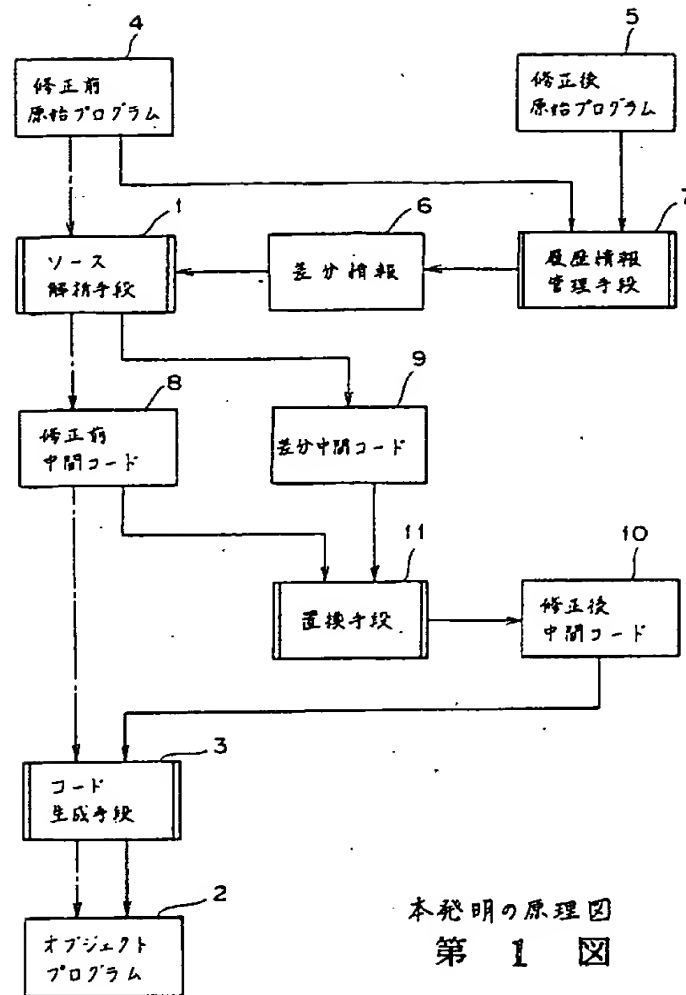
- 1…ソース解析手段、
- 2…オブジェクトプログラム、
- 3…コード生成手段、
- 4…修正前原始プログラム、
- 5…修正後原始プログラム、

- 6…差分情報、
- 7…履歴情報管理手段、
- 8…修正前中間コード、
- 9…差分中間コード、
- 10…修正後中間コード、
- 11…置換手段。

出願人： 富士通株式会社  
代理人： 弁理士 松本 島

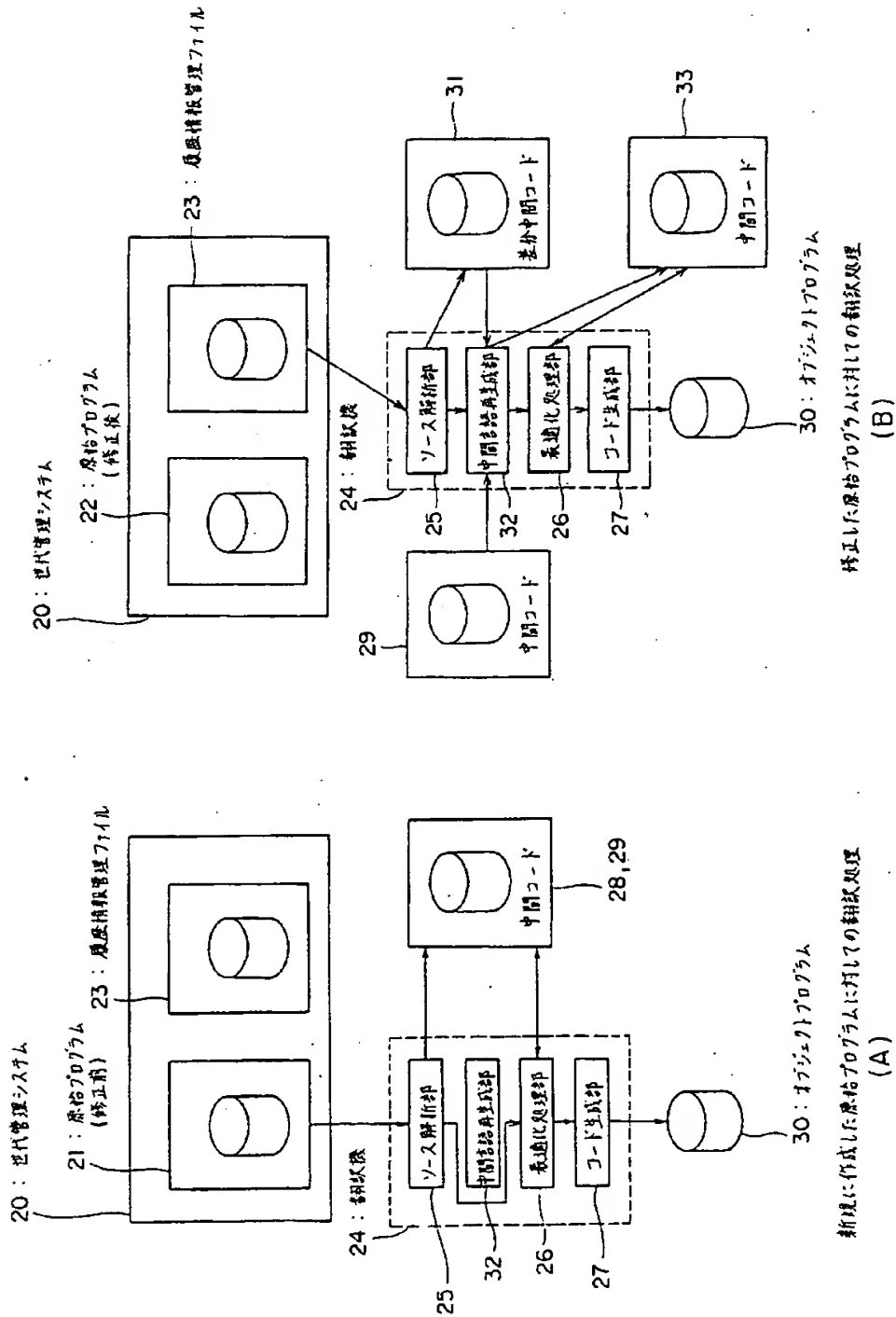
- 17 -

- 18 -



本発明の原理図  
第 1 図





一実施例を示すブロック図  
第 2 図

```

100 IF A = B
200 THEN
300 GOTO LI;
400 FI;
500 C := A;
600 LI: C := B;

```

(A) 入カソース

| テキスト | 行番号 | 符号     | 修正情報 |
|------|-----|--------|------|
| STNO | 100 | A=B L@ |      |
| BNE  |     |        |      |
| STNO | 200 |        |      |
| STNO | 300 | LI     |      |
| B    |     |        |      |
| STNO | 400 |        |      |
| L@   |     | C:=A   |      |
| STNO | 500 |        |      |
| LDST |     |        |      |
| STNO | 600 | C:=B   |      |
| LI   |     |        |      |
| LDST |     |        |      |

| テキスト | 行番号 | 符号     | 修正情報 |
|------|-----|--------|------|
| STNO | 100 | A/=BL@ | A    |
| BEQ  |     | A=B L@ | D    |
| BNE  |     |        |      |
| STNO | 200 |        |      |
| STNO | 300 | LI     | D    |
| B    |     |        |      |
| STNO | 400 |        |      |
| L@   |     |        |      |
| STNO | 500 | C:=A   |      |
| LDST |     |        |      |
| STNO | 600 |        |      |
| LI   |     | C:=B   |      |
| LDST |     |        |      |

(B) ソース解析終了時の中間コード

\* 修正情報 A⇒追加  
D⇒削除

(C) 最適化処理終了時の中間コード

原始プログラム及び中間コードの一例を示す図(修正前)

第 3 図

```

100   IF A/=B ----⇒ 修正箇所
200   THEN
300     GOTO LI;
400   FI;
500   C:=A+1; ---⇒ 修正箇所
600 LI: C:=B;

```

(A) 入力ソース

| テキスト | 行番号 | 符号...   | 修正情報 |
|------|-----|---------|------|
| STNO | 100 | A/=B L@ |      |
| BNE  |     |         |      |
| STNO | 200 |         |      |
| STNO | 300 | LI      |      |
| B    |     |         |      |
| STNO | 400 |         |      |
| L@   |     | C:=A+1  |      |
| STNO | 500 |         |      |
| LDST |     |         |      |
| STNO | 600 | C:=B    |      |
| LI   |     |         |      |
| LDST |     |         |      |

(C) 中間コード再生成処理終了時の中間コード

| テキスト | 行番号 | 符号...   | 修正情報 |
|------|-----|---------|------|
| STNO | 100 | A/=B L@ |      |
| BNE  |     |         |      |
| STNO | 500 | C:=A+1  |      |
| LDST |     |         |      |

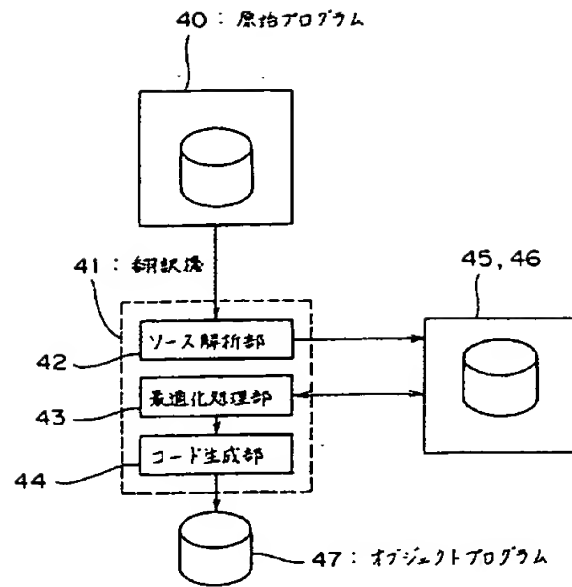
(B) ソース解析終了時の差分中間コード

| テキスト | 行番号 | 符号...   | 修正情報 |
|------|-----|---------|------|
| STNO | 100 | A=B L@  | A    |
| BEQ  |     | A/=B L@ | D    |
| BNE  |     |         |      |
| STNO | 200 | LI      | D    |
| STNO | 300 |         |      |
| B    |     |         |      |
| STNO | 400 | C:=A+1  |      |
| L@   |     |         |      |
| STNO | 500 |         |      |
| LDST |     | C:=B    |      |
| STNO | 600 |         |      |
| LI   |     |         |      |
| LDST |     |         |      |

(D) 最適化処理終了時の中間コード

原始プログラム及び中間コードの一例を示す図(修正後)

## 第 4 図



従来技術のブロック図  
第 5 図

第 1 頁の続き

⑦発明者

横 田

学

福岡県福岡市博多区博多駅前 1 丁目 5 番 1 号 富士通九州  
通信システム株式会社内